Variable-Width Datapath for On-Chip Network Static Power Reduction

George Michelogiannakis, John Shalf
Lawrence Berkeley National Laboratory, 1 Cyclotron Road, Berkeley, CA 94720
Email: {mihelog,jshalf}@lbl.gov

Abstract

With the tight power budgets in modern large-scale chips and the unpredictability of application traffic, on-chip network designers are faced with the dilemma of designing for worst-case bandwidth demands and incurring high static power overheads, or designing for an average traffic pattern and risk degrading performance. This paper proposes adaptive bandwidth networks (ABNs) which divide channels and switches into lanes such that the network provides just the bandwidth necessary in each hop. ABNs also activate input virtual channels (VCs) individually and take advantage of drowsy SRAM cells to eliminate false VC activations. In addition, ABNs readily apply to silicon defect tolerance with just the extra cost for detecting faults. For application traffic, ABNs reduce total power consumption by an average of 45% with comparable performance compared to single-lane power-gated networks, and 33% compared to multi-network designs.

1. Introduction

Large-scale chips, such as chip multiprocessors (CMPs) and graphical processor units (GPUs) are made possible by recent semiconductor scaling. However, due to their large scales, they are increasingly constrained by power [13, 34]. An important component of such large-scale chips is the on-chip network [10]. Even modern and optimized networks contribute significantly to a system's power, area and performance characteristics. For example, the Intel Teraflop chip attributes 26% of its power consumption to the on chip network [21], while MIT RAW attributes approximately 5% in the typical case, with the worst case being 36% [27]. To make matters worse, it is projected that with 2018 technology communication will in fact require more energy than computation, even in the on-chip environment. In addition, communication will increase in order to meet future computation demands [43, 4, 13].

Designing an on-chip network is not independent from the system and applications, although it often has to be performed independently. Variations in application behavior can be important; past work has observed that application demands can vary substantially, and also applications tend to not load the network evenly in both space and time [46, 8, 41, 18, 51, 2, 16]. For example, average injection rates do not exceed 7% for PARSEC benchmarks, although the maximum channel utilization can be 43% [18, 2, 16, 5]. The average channel utilization in a sample PARSEC benchmark can vary from 1% to 15% in different network locations [18]. Similar observations were made for single-threaded SPEC benchmarks [2, 16].

To avoid making the on-chip network a performance bottleneck, the safe choice is to design it to handle worst-case traffic loads. However, this approach creates a network that is over-designed for most applications and most application execution phases. Not only does this increase network area, but it also increases static power. Static power is mainly composed of leakage power and the power to toggle clocking inputs, with leakage typically being the dominant component [33]. The relative magnitude and variance of leakage power is projected to increase in future near voltage threshold technologies [24] or technologies with smaller feature sizes [8, 23].

Past work has demonstrated the importance of leakage power. In the Intel Teraflop chip, leakage power varies from 9.6% to 15.6% [21]. In a larger 256-core system, network leakage power can be 39% of the chip's leakage power even when operating at network saturation [12]. Past work reports that approximately 90% of the network power is leakage with light-traffic application benchmarks, or 30% to 50% with heavy-traffic benchmarks [12, 17, 26]. While the above numbers vary with technology library, implementation, and network parameters, they are clear motivators that on-chip network leakage power is a growing concern given the tight power budgets of modern large-scale chips.

In this paper, we propose adaptive bandwidth networks (ABNs), which activate exactly the amount of bandwidth needed at every channel, and exactly the number of virtual channels (VCs) required at every input port. ABNs accomplish this by dividing channels into lanes. Lanes are activated individually according to local traffic demands. Inactive lanes are power gated, consuming near zero static power.

In addition, we adopt fine-grain power gating in individual VCs similar to [33]. However, unlike past work, we use drowsy SRAM cells which enable ABNs to make activation decisions in the upstream router's VC allocator, thus avoiding mispredictions which can cause activating more VCs than necessary [33]. ABNs also use power gating in router switches by adding multiple lanes for every input and output, similar to [17]. ABNs hide activation delays using a single look-ahead signal per flit for both VC and lane activations, generated at the upstream router after switch allocation [31, 32].

Since channels are divided into lanes and flits can choose different lanes at each hop, ABNs also readily apply to fault tolerance by shutting down only lanes that contain defects, instead of whole channels. This avoids the extra complexity and VCs to enable packet detours [47, 29, 40].

If area is not a concern, with ABNs the proper design choice

is to provide enough network resources for worst-case traffic, thus alleviating designers from a difficult choice. In our experiments, ABNs reduce total power by 15% for uniform random (UR) traffic and up to 45% for application benchmarks with comparable performance, compared to single-lane power-gated networks [31, 32, 45, 46, 8]. Compared to multi-network designs [6, 12], ABNs reduce total power by 33% for application benchmarks and increase throughput by 8% for UR traffic, due to the flexibility flits have to switch lanes in each hop, instead of only at injection time. ABNs also provide better fault tolerance than multi-network designs. ABNs occupy additional area for the power gating logic similar to past work, which has been reported to be 4.3% for routers [32].

2. Background and Related Work

Power-gating techniques typically disconnect cells from power or ground lines in a coarse- or fine-grain manner [32, 48, 42, 8]. This is accomplished by adding high threshold voltage (low leakage) connector transistors. In the context of on-chip networks, power gating typically applies to domains visible to flow control. For example, channels can be activated or deactivated individually, and the network needs to activate channels in time for flit traversal or enable detours and guarantee full connectivity with inactive channels [32, 45, 46, 8]. Power gating of input buffers is possible at the granularity of entire buffers [32, 17], VCs [33], or individual buffer entries [26, 38]. Power gating has also been applied to the switch and allocators [17, 49]. Deactivating entire routers has also been proposed, with the possibility of adding bypass channels such that routers are not activated by traffic that does not need to change directions and under low-traffic conditions [8, 32, 17]. Further related work dynamically scales the voltage or clock frequency of channels and routers [37, 44, 30].

To hide the latency of waking up resources, past work uses look-ahead signals [31, 33]. Look-ahead signals are generated in the appropriate pipeline stage of upstream routers such that channels and VCs will be active by the time flits arrive. However, look-ahead signals can cause false activations if they are eligible to activate multiple resources, such as one of multiple VCs [33]. For example, packet A may be eligible to use one of two VCs in router 1. When the look-ahead signal arrives downstream in router 2, and assuming that another packet B holds one of those VCs, router 2 may conservatively wake up both eligible VCs in anticipation for packet A. That is because packet B's completion time is unknown. Therefore the router may not want to risk stalling packet A because resources were not activated in time. However, if packet B leaves before packet A arrives, activating the second VC was unnecessary since packet A can use the same VC as packet B.

To eliminate false activations but still power-gate VCs, ABNs adopt drowsy SRAM cells. Drowsy SRAM cells were initially proposed for reducing leakage power in caches [14]. The advantage of this technology is that drowsy cells can be activated in a single cycle, and still hold data when drowsy. However, when deactivated, drowsy cells consume more leakage power than power-gated SRAM cells, and require more energy to be reactivated. Drowsy SRAM cells were only briefly investigated in on-chip networks [9].

Past work, related to ABNs, also adjusts channel bandwidth dynamically [18]. This work, however, takes a different approach by using channel wires in a bidirectional manner, in order to increase bandwidth to one direction. Such an approach is orthogonal and therefore applicable to ABNs.

Further related work has proposed networks with configurable bandwidth with the goal of reducing static power, but has done so using multiple subnetworks [6, 12]. In those designs, traffic sources decide whether to inject a packet to an active network or to activate a new network using oblivious [6] or adaptive [12] metrics. Multi- and single-network approaches, such as ABNs, have different tradeoffs, which we explain here and quantify in Section 5.

To begin with, in ABNs, flits are able to use different lanes in each hop. Therefore, flits will use active lanes preferentially over activating inactive lanes, thus minimizing new activations. In contrast, multi-network approaches make decisions for each flit only at injection time. Once injected, flits are unable to switch subnetworks without considerable complexity. Therefore, ABNs reduce the total number of activations as well as the number of cycles channel wires are active for.

Optimal flit placement becomes more challenging under uneven network loading. Consider the case where packets need to be placed in the appropriate subnetwork such that congestion is avoided in a high-traffic region of the network. Those injection choices are not optimal for low-traffic regions. Therefore, if two low-traffic flows are destined to a region of high traffic, where the optimal decision for the network is to place the two flows into separate subnetworks, those flows are unable to share channels in low-traffic regions on their way to the high-traffic region. This is illustrated in Figure 1. In that scenario, ABN simply activates more resources in the high-traffic regions without affecting low-traffic regions.

Flit placement in subnetworks affects performance in addition to energy. Flits encountering congestion are unable to utilize another subnetwork's bandwidth in order to take full advantage of the bisection bandwidth.

In addition, since the choice is made only at injection time in multi-network approaches, deciding adaptively requires waiting for congestion information to propagate to the network endpoints. Knowledge of global and accurate current and future network state is impossible. In ABNs, decisions are made at each router with the most up to date information.

ABNs also more readily apply to fault tolerance since a defect in a single bit of a channel shuts down only the affected lane of that channel. In multi-networks, a single fault would disable an entire subnetwork without the complexity to enable packet detours [47]. Finally, the radix of the network interface at each endpoint increases with the number of subnetworks.

On the other hand, ABNs require larger-radix switches than

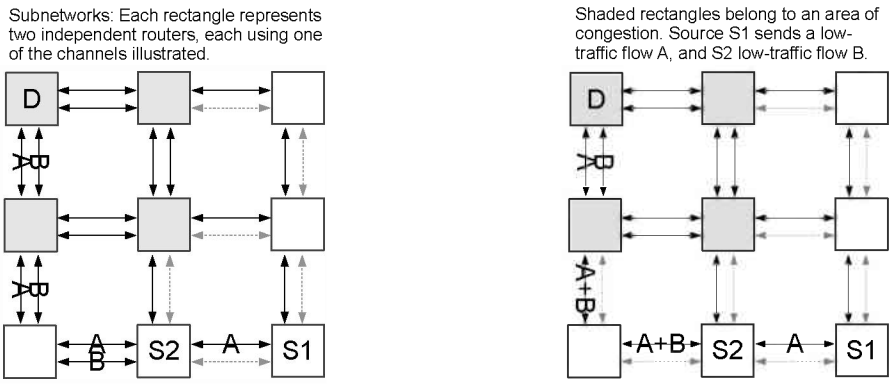

Figure 1: With subnets (left), because flows A and B should be placed in separate resources in high-traffic regions (shaded routers), they are forced to use separate resources in low-traffic regions with multiple and independent subnets. In an ABN (right), packets are free to share resources in low utilization regions. Shaded channels are inactive.

multi-network approaches. Comparing an ABN with two lanes with the same bisection bandwidth as a multi-network design with two subnetworks, the ABN has half the number of switches but they have twice the radix each. Due to the quadratic cost of switches with radix [35], this results in approximately half the switch area and energy for multi-networks compared to ABNs, as well as simpler switch allocators.

Mechanisms to detect silicon defects during network operation have been proposed [15]. Faulty channels can be disabled which forces packets to use alternate and often longer paths [29, 40]. To increase a channel's tolerance to silicon defects, spare channel bits can be associated with every channel [10, 52, 25]. Other mechanisms use channels with defects partially by serializing and deserializing flits [39, 7]. Past work also uses bidirectional channels to avoid costly detours in the presence of faults [47]. ABNs achieve a similar goal, but take a more graceful approach by using a single mechanism to both increase fault tolerance and reduce static power.

ABNs advance the state of the art by proposing an architectural technique to take advantage of the low activation latency of drowsy SRAM cells in order to avoid false activations when power gating individual VCs, varying the amount of bandwidth in every hop to match traffic demands but without the drawbacks of multi-network approaches analyzed above, and gracefully using the same lane and VC activation mechanism for channel silicon fault tolerance as well.

3. Adaptive Bandwidth Networks

In ABNs, channels are divided into lanes, as shown in Figure 2. Each lane is a power gating domain that is activated independently to match traffic demand. VCs in input buffers are also activated independently [33]. At every hop, flits are free to choose a different lane and VC. Finally, router switches are also divided in power gating domains (lanes) [17]; each switch input port has as many lanes as input buffers have VCs, and each switch output port has as many lanes as channel lanes.

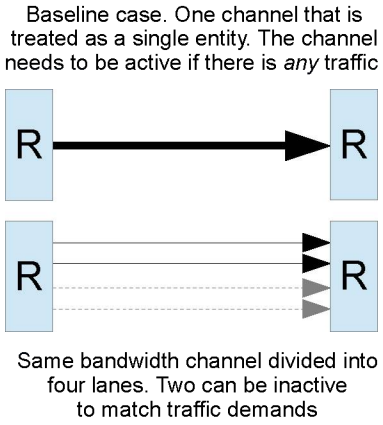

Figure 2: Channels are divided into lanes. Each lane is an independent power gating domain.

3.1. Multi-lane Channels

Figure 2 illustrates the basis of ABNs. Channels are divided into lanes without affecting the bisection bandwidth (increasing the bisection bandwidth is an option always available to the designer and orthogonal to this work). Each lane is an independent power gating domain. In the example shown, two lanes are active and two are inactive. However, any number of lanes can be active or inactive to match traffic demands.

We use the channel power gating models of [32, 9], which disconnect cells from ground using high voltage threshold (low leakage) connector transistors. In those models, the activation latency ($Lane_{ActLat}$) has been reported to be no more than 3ns in a 65nm process. Power-gated channel bits consume 0.5% of their leakage power ($Lane_{inact}$), due to the connector transistors. Channel lanes are activated by the router that is driving data on them. Therefore, each lane requires an extra bit to control the lane's status. That bit controls the high voltage threshold connector transistors. Finally, the activation energy penalty is the equivalent of eight clock cycles of leakage power at 1GHz ($Lane_{ActPen}$). This covers the activation penalty, the propagation of the control bit, as well as the gradual increase

and decrease of leakage power during the activation and deactivation periods. This means that the break-even time, which is the number of cycles the lane must remain inactive for to cover the activation penalty, is eight cycles.

We restrict packets to using one lane per hop per cycle. In other words, multiple flits of the same packet may not be transmitted in the same cycle using different lanes. This decision was made in the interest of static power and resembles the operation of alternative techniques such as multi-networks [6, 12] which assign packets to a single subnetwork. Without this restriction, a packet could activate all channel lanes, which defeats the purpose of channel lanes. However, this restriction increases serialization latency similar to multi-networks, compared to the baseline network of Figure 2. Even though execution time never increases more than 1% with two-lane ABNs for our application benchmarks, increasing serialization latency can be a drawback in latency-sensitive systems. In those cases, the network can submit flits of the same packet in the same cycle in multiple lanes, similar to [50].

3.2. Router Datapath

For input buffers, we use drowsy SRAM cells following the models of [9]. Each VC can be activated and deactivated independently from other VCs and in a single cycle (VC_{ActLat}). When inactive, drowsy SRAM cells consume 15% of their active leakage power (VC_{inact}). We pessimistically model the energy penalty for activating a VC to equal sixteen cycles of leakage power at 1GHz (VC_{ActPen}). This covers both the energy penalty of the activation and the leakage power during the cycle that a VC is activating or deactivating. Therefore, the break-even time in this case is sixteen cycles.

Even though channels may deliver one flit per lane per cycle, those flits are guaranteed to be destined to different VCs because packets are restricted to send only one of their flits per cycle. Therefore, VCs have the same width as channel lanes and it is possible to place VCs in separate SRAM blocks with a single read and a single write port each. This implementation makes it straightforward to manage each VC as an independent power gating domain.

To avoid making the input side of router switches a bottleneck, router switches need to connect to every input VC with separate switch lanes. This enables flits from any input VC to select any output lane. At the output side, switches connect to each lane of each output channel. Therefore, switches have $(InputPorts \times LaneWidth \times VCs)$ input bits and $(OutputPorts \times LaneWidth \times ChannelLanes))$ output bits, because VCs have the same width as channel lanes. This way, routers can transmit a flit to every lane of every output port in each cycle, and each input VC can also transmit a flit independently of other input VCs. Compared to a single-lane network with the same bisection bandwidth, the output side of ABN switches have the same number of bits, whereas the input side's number of bits is no greater than the single-lane network as long as $ChannelLanes \leq VCs$.

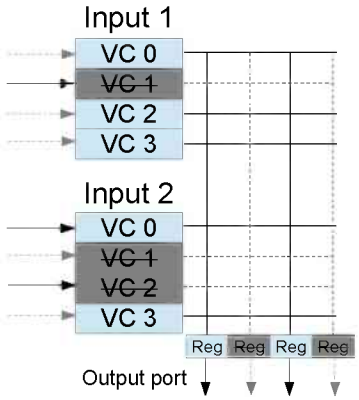

Figure 3: Only two inputs and one output are shown. The switch connects to each VC and each output lane. Those connections are power gated according to the state of the VCs and output lanes, as shown.

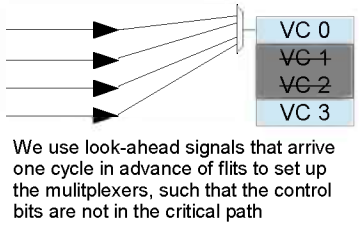

Figure 4: A multiplexer for each VC is required because flits from any lane may be destined to any VC.

ABNs apply power gating to switches to reduce static power [17], similar to channels. This is the third power gating domain of ABNs. At the input side, the switch connection to a VC is only active when the VC is active. At the output side, switch lanes are only active when the corresponding channel lane is active. The router datapath is illustrated in Figure 3.

At the input port side, buffers face a potential source of extra complexity because incoming flits in any lane may be destined to any input VC. This requires a simple switching fabric such as a multiplexer for each input VC, implemented with two or three levels of AND and OR gates. Each input VC's multiplexer connects to every lane of that input channel, as shown in Figure 4. To mitigate the potential timing overhead to the last pipeline stage of the channel, we guarantee that the control bits which set the multiplexer at each input VC arrive in the cycle prior to the flit's arrival, using the look-ahead signal for that flit. This is an additional function the look-ahead signal serves (explained in Section 3.3), in addition to activating lanes and VCs. This way, multiplexer setup is not in the critical path since the control bits are supplied at the beginning of the cycle after the arrival of the look-ahead signal, while the flit arrives at the end of that same cycle.

Still, with a large number of channel lanes, this switching fabric may pose a noticeable timing overhead on the last pipeline stage of input channels even with the control bits preset. We can simplify this switching fabric by restricting the freedom flits have in choosing lanes. For example, in a net-

work with an equal number of lanes and VCs, we can enforce a direct correspondence between lanes and VCs, such that when flits choose a VC they also implicitly choose a lane. Restricting the choices in lanes flits have reduces the size (radix) of the multiplexers for every input VC and simplifies switch allocation since flits are no longer eligible for all output channel lanes. Networks with an unequal number of lanes and VCs can map a subset of VCs to each lane. While this technique will result in more channel lane activations than necessary due to choice restriction, it also simplifies switch allocation by reducing the possible input VC–output port combinations. Finally, in the case that the number of VCs equals the number of lanes, reserving an output VC essentially also reserves a lane in the output channel. Therefore, the VC allocation step is no longer required, and routers can have one less pipeline stage. We call this option *ABN simple* and quantify its efficiency in Section 5. Since packets choose a VC before lanes, both the restricted and unrestricted schemes use VCs similarly to networks without lanes for deadlock avoidance.

3.3. Router Pipeline and Control

The router pipeline is illustrated in Figure 5. For each flit receiving a switch allocator grant, a look-ahead signal is generated to alert the downstream router of the impending flit arrival. Because flits have to traverse the switch in the next cycle, flits and their corresponding look-ahead signals are separated by one cycle. Look-ahead signals are generated for each flit, even if from the same packet as an older flit. That is necessary because switch allocation does not have to respect packet length, and may thus delay subsequent flits long enough for resources to power down. Each look-ahead signal contains:

- The output the flit will request in the downstream router, obtained in the upstream router using look-ahead routing [31].
- The input VC in the downstream router (output VC in the upstream router).

Therefore, the number of bits for the look-ahead signal are $\log_2 Output ports + \log_2 InputVCs + ValidBit$. One such set of bits for look-ahead signals is required per channel lane. Look-ahead signals serve two functions:

- They alert the downstream input buffers of the VC the flit will be arriving into. This is used to activate the appropriate VC buffer and to set up the multiplexer at the input side of that VC. Activating a VC buffer also activates the downstream switch input lane that connects to it.
- They alert the downstream router of the output port that flit will request. This is used to activate lanes at that output channel as well as switch output lanes to that output.

Channel and switch lanes are activated according to the number of flits that are destined to each output port. Routers maintain a counter per output port. Look-ahead signals increment the counter for the appropriate output port. Flits receiving a grant in switch allocation and departing the input buffers decrement the counter. Routers activate as many channel lanes for each output and the corresponding switch output lanes to match the counter's value, but with a delay of $Lane_{ActWait}$ cycles. Specifically, lane X is activated if the counter's value for that output has been at least X continuously for the last $Lane_{ActWait}$ cycles. This ensures that ABNs do not over-react to short-lived congestion that resolves with existing bandwidth after a few cycles. For example, if two flits arrive for the same output port in the same cycle, those flits can share the same lane with only a clock cycle's latency penalty to one of them. Using $Lane_{ActWait}$ does not appreciably affect steady-state traffic performance, but assists in reducing leakage power under dynamic traffic patterns. Dynamic lane activation policies that consider credit count or other network state, as well as historical information, are left for future work. Multi-network approaches do not have this flexibility because flits may not switch to an active but idle subnetwork.

Lanes are deactivated after a predetermined number of cycles of inactivity ($Lane_{DeactWait}$). Routers are responsible to deactivate lanes in their output channels.

Since look-ahead signals are generated before switch traversal and activate lanes in the downstream router, they can hide three cycles of activation latency for switch and channel lanes. In our 65nm commercial technology library and lane power gating models [32, 9], this is enough to hide the lane activation delay in full. In addition, since switch traversal precedes link traversal by a cycle, channel lanes need only be activated one clock cycle after the corresponding switch lanes.

Because of the proactive nature of lane activation, false lane activations are possible. A false lane activation is an activation of a switch and channel lane without subsequent flit traversal. Consider the case where flits A and B, belonging to different packets, arrive in the same input for the same output port. A is stalled because its VC has no credits. B, however, is able to make progress and departs the router before A. A then departs the router in the future. In this example, two lanes were activated to guarantee that A and B will not stall waiting for lanes. However, due to credits, only one flit departed the router at a time. Therefore, one active lane would suffice. We quantify the frequency of false lane activations in Section 5.1.

On the input buffer side, the look-ahead signal arrives only one clock cycle in advance of the flit. One cycle suffices to hide the single-cycle activation delay of drowsy SRAM cells and to drive the control bits of the multiplexers at the input side of input buffers at the beginning of the next clock cycle, using a pipeline register. Power-gated (non-drowsy) SRAMs which require three cycles of activation in similar technologies as our models [33] would pose no latency penalty because we could generate an additional look-ahead signal before VC allocation in the upstream router's pipeline to hide the delay. However, using drowsy SRAM cells with their single cycle activation delay enables the use of a single look-ahead signal per flit to serve all functions, thus reducing overhead. More importantly, the single-cycle activation delay enables the use of the upstream router's VC allocator to decide what VCs to activate downstream. This eliminates the possibility of false

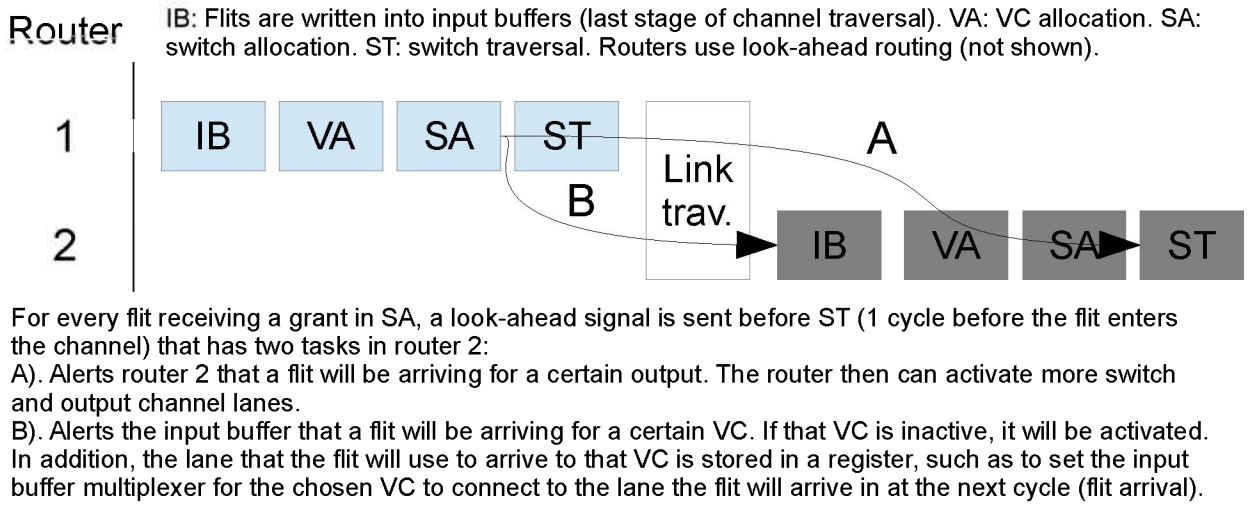

Figure 5: The pipeline for two consecutive routers. A look-ahead signal is created for each flit winning switch allocation.

VC activations [33] because only VCs that flits are actually assigned to are activated. We quantify the effect of false VC activations as a motivator for drowsy SRAMs in Section 5.1.

To reduce unnecessary VC activations, VC allocators prioritize active output VCs. Even though flits are assigned a VC during VC allocation, the downstream VC is not activated until after the flit wins switch allocation. This is another advantage of drowsy SRAMs, since power-gated SRAMs may be activated by flits which are later delayed during VC or switch allocation. Input VCs are deactivated if empty and idle for $VC_{DeactWait}$ cycles. Input VCs are also deactivated if empty and all channel lanes to that input are inactive.

3.4. ABN Complexity

ABNs increase switch allocator complexity because multiple grants may be generated for each output (one for each lane), and each input VC may be granted independently of other VCs of the same input. With ABNs, switch allocators have to perform an $(InputPorts \times VCs) \times (OutputPorts \times ChannelLanes)$ allocation, where the same input can receive multiple grants to different VCs in the same cycle. However, in the typical case where $ChannelLanes \leq VCs$, the switch allocator is no more complex than the VC allocator, which performs an $(InputPorts \times VCs) \times (OutputPorts \times VCs)$ allocation, where the same input can also receive multiple grants to different output VCs. Past work has analyzed the impact of radix in both VC and switch allocators and reports that in a typical mesh with 2 VCs, extending the radix of the switch allocator to become equivalent to that of the VC allocator extends the switch allocator's minimum timing path by 10% for separable allocators [3]. Even in a high radix flattened butterfly (FBFly) topology [28], the switch allocator's path is only extended by 15%. However, given than VC and switch allocation are typically performed in separate pipeline stages and the switch allocator is no more complex that the VC allocator if $ChannelLanes \leq VCs$, this timing overhead is unlikely to extend the router critical path.

In addition, increasing the radix of the switch allocator in a mesh with 2 VCs to match that of the VC allocator would increase area by approximately 30% and power by 35% for separable allocators [3]. However, the VC allocator occupies approximately 5000 um^2 and consumes 2 to 10 mW, both of which are very small percentages of the router [3, 23]. Therefore, the area and power increase of the switch allocator will have a marginal impact to the router. This is confirmed by the Intel Teraflop chip which consumes 7% of the network's power for allocation and *all other router logic* [21].

Networks without VCs may experience an increase in the router's critical path if that path includes allocation, because such networks have no VC allocator to hide the extra delay of the switch allocator with pipelining. In such networks and if cycle time is a priority, this factor needs to be evaluated as a tradeoff with the power and performance benefits of ABNs over alternative approaches, explained in Section 5.

As stated in Section 3.2, ABNs do not increase router switch radix compared to a single-lane network with the same bisection bandwidth, as long as $ChannelLanes \leq VCs$. The same Section also discusses the simple extra logic in the input side of router buffers in ABNs, and how to simplify it. Moreover, ABNs have the same overhead in channels and switches compared to past work with power gating, because ABNs use the same models with power-gated transistors [17, 32, 48, 42, 8, 6, 12]. Finally, past work on power-gated networks also use look-ahead signals to cover resource wake up time, similar to ABNs [31, 33]. Look-ahead signals increase channel dynamic energy by 2% in our experiments for all power-gated networks.

3.5. Silicon Defect Tolerance

While the primary purpose and novelty of ABNs lie in static power consumption, ABNs also readily apply to making use of channels with defects, with only the additional cost for

detecting faults. With ABNs, faulty channels, switch lanes, or input VCs can simply be disabled, but the remaining resources are still usable. Therefore, for a single fault, there is no need to disable whole input ports or channels and resort to extra VCs and more complex routing algorithms that enable detours [29, 40]. If we define the fault probability for a single channel bit P (ranging from 0 to 1), channel width W, number of lanes L, the probability for a channel to fully fail is:

$$(\frac{W}{L} \times P)^L$$

$L = 1$ represents the baseline single-lane network and assuming channel bit failures are independent events. Lanes are considered failed if they contain at least one faulty bit.

As the number of lanes increases, the probability that all lanes contain a fault decreases. Therefore, ABNs are more likely to maintain network connectivity with an equal number of faults compared to the baseline single-lane network. Multi-network designs will fail if a single channel in any subnetwork fails, without extra VCs and complexity to enable detours [47, 29, 40]. That is because flits already injected to the faulty subnetwork cannot switch subnetworks, and there is propagation delay to alert all traffic sources of the fault.

4. Methodology

For our evaluations, we use a modified version of Booksim [22]. We present results for synthetic traffic and PARSEC benchmarks collected with Netrace traces [5, 19], which respect packet dependencies and therefore reflect the impact of the network to application execution time, in contrast to timestamp-based traces. For synthetic traffic, we use UR, bit complement, tornado, and hotspot traffic where all sources send to a single hotspot destination [11]. For synthetic traffic we use a read-reply communication protocol. The traffic pattern decides the destination of read and write requests. Each request generates a reply back to the request's source. Read requests and write replies are 128 bits. Write requests and read replies are 640 bits. For our synthetic traffic we vary the injection rate of request packets.

For PARSEC simulations, we use the Netrace traces provided in the project's website which were collected for a 64-core cache-coherent CMP. The processors are in-order ALPHA cores at 2GHz. L1 caches have 32KBs for instruction and 32KBs for data. They are 4-way set associative and use MESI cache coherency. L2 caches are fully shared S-NUCA with 64 banks and 16MB, eight-way set associative and 8 cycle bank access time. Finally, the memory has a 150-cycle access time and 8 on-chip memory controllers. We simulate 200,000 packets of the application's parallel execution region. We simulate seven benchmarks using their medium size input sets.

We compare the following networks:

- *Baseline network without power gating (baseline)*: This network illustrates the baseline case without power gating.
- *Single-lane power gating network (single lane)*: This network represents the state of the art in single-network power gating [31, 33]. In this network we still use drowsy SRAM cells to prevent false activations and fully hide VC activation latency [33, 9]. We make this choice in order to isolate the gains from channel and switch lanes.
- *Flexible adaptive bandwidth network (ABN flexible)*: This is the network proposed in this paper. We keep bisection bandwidth constant compared to other networks because increasing bisection bandwidth is orthogonal to this work. Therefore, with two lanes, flits are half the width compared to the networks above, and each input port has twice the VCs. Flits can traverse from any lane to any VC.
- *Simple adaptive bandwidth network (ABN simple)*: Same as above, but we map lanes to only allow delivery to a subset of VCs, for the reasons explained in Section 3.2. With four VCs and two lanes, each lane delivers to one request VC and one reply VC. Since a single lane maps to multiple VCs, both VC and switch allocators are required.
- *Multi-network designs (multinets)*: This network represents the state of the art in static power reduction and consists of multiple subnetworks, each of which operates similar to the single-lane power gating network [6, 12]. Sources inject flits to the first subnetwork unless congestion is sensed in the injection router by a count of available buffer space in all input buffers of the injection router [12]. If the available buffer space is less than half of total buffer size, sources consider the next subnetwork, and so on. If all subnetworks are congested, sources choose one at random. Bisection bandwidth is equal to ABNs, and therefore flit width is also equal as long as the number of lanes in ABNs is the same as the number of subnetworks.

We use an 8×8 2D mesh with dimension-order routing (DOR) and 2mm channels. The baseline single-lane network has 128-bit channels and two VCs per input. VCs are equally divided across requests and replies. To keep total buffer size constant for all networks, we increase the number of VCs for networks with more than one lane, because such networks have narrower flits and therefore VCs. We choose to increase the number of VCs because keeping the number of VCs constant but making them deeper typically does not justify the increased cost as long as the credit round-trip delay is covered. Therefore, we provide four VCs per input port in two-lane ABNs, instead of two VCs in single-lane networks. Multinets also have more VCs in total but the VCs are distributed among subnetworks. By default, we present ABNs with two channel and switch lanes, and multinets with two subnetworks. We use the router pipeline of Figure 5.

For cost estimation, we use a commercial 65nm technology library and the area and power models of [1]. These models use custom SRAMs for buffers, which have lower leakage and dynamic power consumption than flip-flop (FF) arrays or compiler-generated SRAMs, frequently used in past work [26, 9, 17, 8]. Our SRAM models have been verified

Table 1: Network and model parameters.

Parameter	Value
$Lane_{ActLat}$	3 cycles
VC_{ActLat}	1 cycle
$Lane_{inact}$	0.5% of full leakage
VC_{inact}	15% of full leakage
$Lane_{ActPen}$	8 cycles worth of leakage
VC_{ActPen}	16 cycles worth of leakage
$Lane_{ActWait}$	15 cycles
$Lane_{DeactWait}$	3 cycles
$VC_{DeactWait}$	6 cycles
$Area_{Overhead}$	7%

against HSPICE [36]. To estimate the impact of power gating in channels, switches, and buffers, we use the models of [32, 9, 17]. From these models, we pessimistically estimate the power gating area overhead to be 7% for buffers, switches and channels ($Area_{Over}$). Drowsy SRAMs can be activated in a single cycle [14], whereas for channel and switch lanes the activation latency has been reported to be no more than 3ns in a 65nm process [32]. This translates to three clock cycles with our 1GHz clock frequency and 65nm library. Since ABNs can hide three cycles of lane activation delay, ABNs fully hide the channel lane activation delay in this technology process. We derive the configuration parameters shown in Table 1 based on our models and preliminary evaluations. $Lane_{ActPen}$ and VC_{ActPen} are given in the number of cycles that produce the equivalent leakage energy. While these parameters depend on the probability of flits reusing lanes or VCs, which depends on the traffic pattern, we choose one set of numbers for all traffic patterns to simplify design.

We report static power which is predominantly composed of leakage power but also includes the power to toggle the capacitance of the clock input of cells and SRAMs. To simplify comparison between networks, static power also includes energy penalties from activating resources. Dynamic power consists of the power for flits to traverse the network, as well as the power for look-ahead and wakeup signals. Our models do not include allocator and routing logic power, but that has been reported to be a minor contributor [21]. Our area models include input buffers, channels, switches and output registers.

5. Evaluation

5.1. Synthetic Traffic

Results for the mesh under UR traffic are shown in Figure 6. ABN simple saturates at an 8% higher injection rate than multinets because in multinets flits cannot escape congestion encountered at a subnetwork, and perfect injection decisions are unrealistic. Even in ABN simple flits can switch lanes by being granted another VC, since in our simulations we have two VCs assigned to each lane. This also causes multinets to have a 34% higher average latency close to saturation (40% request packet injection rate). Baseline and single-lane have comparable performance, but they each provide an 8% lower throughput than ABN simple because ABN simple has twice the number of VCs and therefore the effects of head of line blocking are reduced. However, due to serialization latency, baseline and single-lane have a 10% lower average zero-load packet latency compared to ABNs and multinets.

ABN flexible and multinets have comparable power consumption across injection rates. However, ABN simple has a 3% higher power consumption because it has less flexibility and thus activates more lanes than ABN flexible. Multinets and ABN flexible have a 15% lower power consumption than the single-lane network, and 24% compared to the baseline.

Figure 6 also separates static and dynamic power. Multinets has 7% lower dynamic power compared to ABN simple and flexible as well as single-lane and baseline. This is because multinets uses switches of half the radix, which therefore incur one quarter of the cost each. This is an inherent property of dividing a single network into multiple subnetworks that multinets takes advantage of [35]. However, because flits pick a subnetwork at injection time with imperfect knowledge and also are not able to switch subnetworks in order to avoid activating a new lane when there is an idle active lane at another subnetwork, multinets has 9% higher static power compared to ABN flexible. This is despite the false activations ABN flexible experiences, and offsets the gains in dynamic power for multinets in our experiments. ABN simple has no false lane activations because once a packet chooses an output VC, all flits have to use the lane that output VC is assigned to. ABN simple and multinets have comparable channel and switch lane activation power overheads because of their comparable number of activations. ABN flexible has a 19% lower activation power overhead because it experiences 17% fewer activations, since it is free to make maximum use of already active lanes.

Comparing ABN simple and flexible, ABN simple saturates at a 21% higher injection rate than ABN flexible. This is because in ABN flexible flits may request any output lane. This increases the allocation problem and intensifies the inefficiencies of our separable single-iteration VC and switch allocators. In contrast, ABN simple restricts the number of lanes flits request and thus significantly simplifies the allocation problem in each cycle. We confirmed this hypothesis by using an augmenting paths allocator which made all networks saturate at comparable injection rates. However, we still use separable allocators because augmenting paths is infeasible in the tight cycle time constrains of the on-chip environment [20].

To further focus on the tradeoffs between the different networks, Figure 7 presents a power breakdown for the mesh under a 2% request packet injection rate. Dynamic power is the power for flits to traverse the network, while static power is predominantly composed of leakage power but also includes the power to toggle the capacitance of the clock input of cells and SRAMs (this is approximately 20% of total static power).

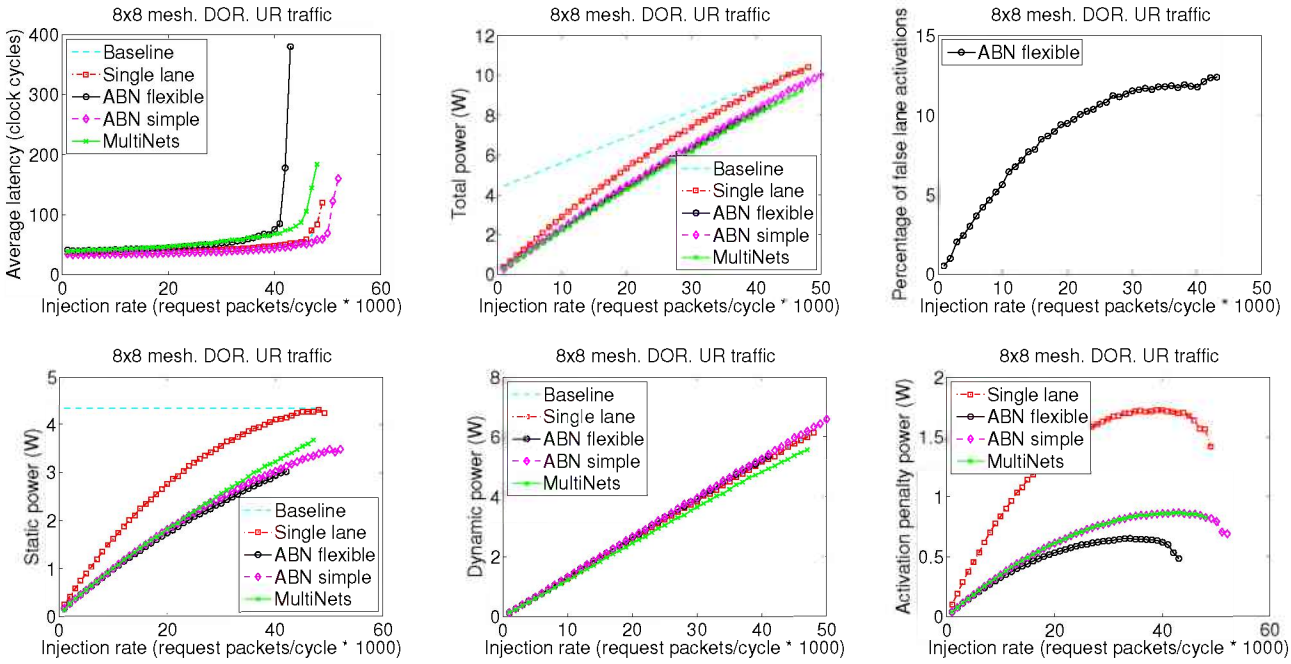

Figure 6: Performance, power and false lane activations for the mesh with UR traffic. ABNs have two lanes and multinets consist of two subnetworks. Static power includes activation penalty power. ABN simple and multinets have no false lane activations.

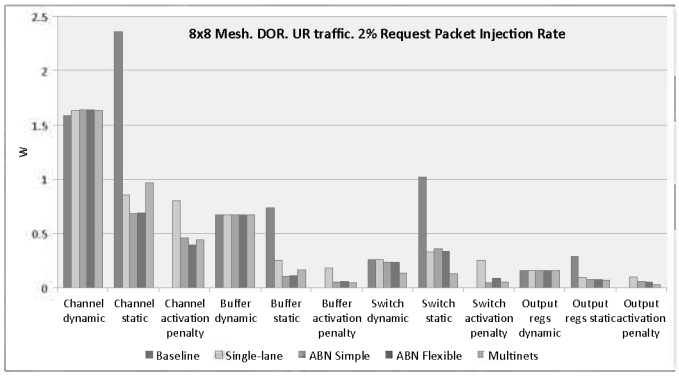

Figure 7: Power breakdown for the mesh. Static power does not include activation penalties (shown separately).

Channel leakage power includes leakage from pipeline FFs and repeaters. Our models use custom SRAMs for buffers [1], which have lower leakage and dynamic power consumption than compiler-generated SRAMs or FF arrays, often used in past work. Our SRAM models have been verified against HSPICE [36]. Future or near voltage threshold technologies may increase the importance of leakage power [8, 23, 24].

As shown, all power-gated networks have a 2% higher channel dynamic power compared to baseline due to the signals to activate resources. Multinets has a 36% larger channel static power (only in this plot static power does not include activation penalties) than ABNs because multinets cannot merge two low-load flows in different subnetworks into a single active lane, as explained in Figure 1. For the same reason, multinets has a 48% larger buffer static power. Multinets has approximately half the dynamic switch power and 61% lower static switch power compared to ABNs because it has twice the number of switches, but each switch is half the radix of an ABN switch. The breakdown includes power for the registers which connect the switch to output channels.

Simulations with the other synthetic traffic patterns we describe in Section 4 show similar and higher performance gains for ABNs (up to twice the saturation rate for hotspot), and up to 3% total power gains for ABNs compared to multinets. However, our synthetic patterns, while a useful tool for an analysis of tradeoffs, hardly exhibit the imbalance in space and especially time of realistic applications, in which the flexibility of ABNs in choosing lanes in each hope is beneficial for compared to multinets. As discussed in Section 5.2, ABNs provide up to 33% lower total power compared to multinets for application traffic.

To motivate our choice of drowsy SRAM cells, we compare ABN flexible with drowsy SRAMs and power-gated (non-drowsy) SRAMs using the models of [33, 14] and UR traffic. At low loads, we observe an average of 43% false VC activations with power-gated SRAMs (there are no false activations with drowsy SRAMs), and 15% more active VC cycles for power-gated SRAMs. However, due to the different energy overheads, non-drowsy SRAMs consume 22% less activation power, but 40% more static power due to the extra active cycles, resulting in 13% higher static power (including activation overhead) overall. At high loads, static power without activation overheads was comparable, but non-drowsy SRAMs incur 12% more activation power due to the 35% false VC activations. While these numbers depend on the number of

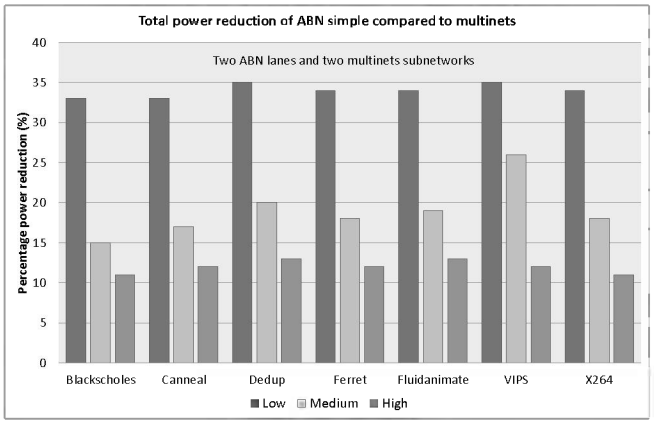

Figure 8: Total power reduction of ABN simple with two lanes compared to multinets with two subnetworks.

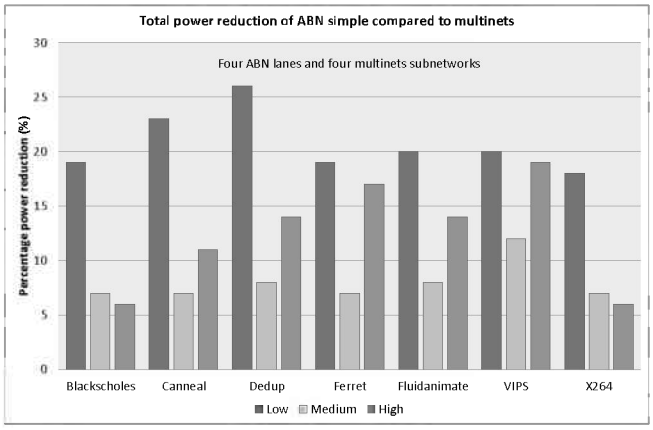

Figure 9: Total power reduction of ABN simple with four lanes compared to multinets with four subnetworks.

activations and active cycles, which depend on the traffic pattern, they show the benefits that drowsy SRAMs can provide, along with simplifying the router pipeline since one wakeup signal suffices for both VCs and channel lanes, as explained in Section 3.3. The single-cycle activation delay of drowsy SRAMs also permit hiding the VC activation delay even with very shallow router pipelines. Finally, since drowsy SRAM cells hold data when drowsy [14], there is opportunity to permit all buffer entries be drowsy at all times except the entries that the head and tail pointers point to; this option should be carefully evaluated against the additional activation energy.

5.2. Application Traffic

For our PARSEC simulations, we first simulate the traces and respect packet dependencies. This produces very low loads to the network, with approximately a 0.2% flit injection rate across benchmarks. We call this the low load testcase, which also evaluates application execution time. We then relax packet dependencies and increase the flit injection rate to 2% and 3.5%, by average across benchmarks. This traffic pattern is not used to measure execution time, but rather is used to load the network in a manner closer to an application with higher loads than our PARSEC benchmarks, and more realistic load distribution in time and space than synthetic traffic. We call the former testcase medium load and the latter heavy load.

Figure 8 presents the percentage of total power reduction of ABN simple with two lanes compared to multinets with two subnetworks. We observe a significant total power reduction for low loads of approximately 33% for ABN simple. Compared to the synthetic traffic patterns shown earlier, application traffic is often bursty and can create hotspots [46, 8, 41, 18, 51, 2, 16]. Hotspots exacerbate the impact of the lack of flexibility flits have in choosing subnetworks after injection, as shown in Figure 1. Bursty traffic also creates temporary congestion in routers which causes flits injected to that router to be sent to the second subnetwork. Even though that is the correct decision for the injection router, flits may not switch subnetworks after traversing the injection router,

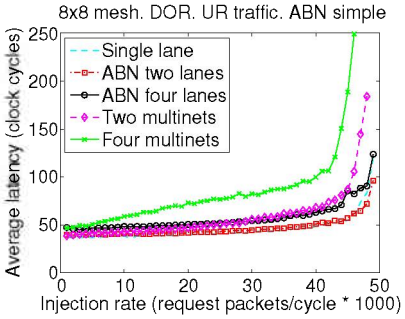

Figure 10: Scaling of ABNs and multinets with UR traffic.

and therefore may not share active resources with other low traffic after injection. This results in 43% to 55% more active cycles by average across benchmarks for channel and switch lanes for multinets compared to ABN simple. Compared to single-lane power-gated networks, ABN simple reduces total power by an average of 45%. Both ABNs and multinets cause an average slowdown of just 0.95% due to serialization latency, with the maximum being 1.05% in the case of blackscholes.

5.3. Increasing the Number of Lanes

In this Section, we explore the effects of dividing the same bisection bandwidth to four lanes for ABNs and four subnetworks for multinets. As shown in Figure 9, ABN simple retains significant total power reductions of approximately 21% under low loads, 8% under medium loads, and 13% under high loads, compared to multinets with four subnetworks. Power reductions are smaller compared to Section 5.2 because further subdividing into more subnetworks makes router switches more energy efficient due to their quadratic cost with radix. In addition, power gains under high loads are larger for ABN simple compared to medium load because the lack of flexibility of flits in multinets becomes more pronounced. ABNs also have a marginal (1%) benefit in execution time under low loads in five benchmarks compared to multinets.

To better understand the effect in performance a variety of loads, Figure 10 presents latency under UR traffic. As

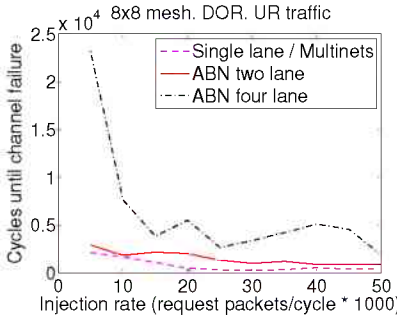

Figure 11: Time until a packet chooses an output port leading to a failed channel, for single-lane and ABN flexible.

shown, multinets with four subnetworks saturates at a 5% lower injection rate and has a 45% higher average latency compared to ABN simple with four lanes. This is because flits encountering congestion in a subnetwork are unable to switch to other subnetworks which may be idling. In contrast, flits in ABN simple can switch lanes by choosing different VCs. Finally, ABNs with four lanes and multinets with four subnetworks have a 19% higher average zero-load latency than ABNs with two lanes and 26% higher compared to the single-lane network.

Therefore, increasing the number of lanes decreases the total power gains of ABNs compared to multinets. However, increasing the number of lanes increases the performance benefits of ABNs compared to multinets, as shown by application execution time and packet latency. Even with lower power benefits, ABNs reduce total power by 21% for application benchmarks. Scaling ABNs to more lanes increases serialization latency similarly to multinets and does not significantly affect throughput. As discussed in Section 3.2, as long as $ChannelLanes \leq VCs$, the switch allocator remains no more complex than the VC allocator.

5.4. Silicon Defect Tolerance

To practically measure the improved resiliency of ABNs to channel bit errors, we simulate UR traffic with varying injection rates, but assign a 5×10^{-4} probability that any one channel bit line will fail in each cycle. This probability is unrealistically high, but we do this in the interest of practicality of our simulations. We also assume that channels have two spare bit lines [10, 52, 25]. We report the time that a packet chooses an output port that leads to a failed channel (without detours). In the case of ABN flexible, this means all lanes have failed. Essentially, this is the time period that the network is no longer able to function correctly. Multinets has comparable probability as the single-lane network regardless of the number of subnetworks, because when a channel in any subnetwork fails, flits already injected may not switch subnetworks to avoid the faulty channel, and there is propagation delay to alert sources.

Figure 11 show the increased resiliency of ABNs. Despite the noise in our results stemming from the many random choices in each experiment, the time until a packet chooses a failed channel is reduced with the increase of injection rate because at higher loads there are more packets requesting outputs. Non-UR traffic that uses only a subset of the channels would ignore failures in parts of the network. ABN simple performs in between ABN flexible and multinets, which reflects the lane selection flexibility they provide to flits.

6. Discussion

Our results illustrate the advantage of providing flits the flexibility to switch lanes in ABNs with local per-hop decisions to accommodate regions with different traffic conditions, compared to multinets where the decision is made once at injection time where perfect knowledge of current and future global state is impossible. This translates to throughput and latency benefits because traffic can be better load balanced across the bisection bandwidth, as well as energy benefits as explained by our results and the example of Figure 1. However, dividing a single network into subnetworks with multinets makes router switches more energy and area efficient. In addition, if $ChannelLanes > VCs$, the switch allocator in ABNs is more complex than the VC allocator and can therefore affect timing. Finally, compared to ABN flexible, ABN simple performs better because its allocators are more efficient, but has less flexibility in assigning lanes, which results in loss in energy.

Our results depend on the relative contributions of channels and switches. Topologies with higher-radix switches, such as the FBFly [28], favor multinets because router switches have a higher radix, and thus dividing into subnetworks in multinets would produce larger power reductions. In contrast, topologies with longer channels, such as a mesh with longer channels than our mesh, favor ABNs because ABNs reduce channel leakage power compared to multinets. The traffic pattern can also favor ABNs if it exacerbates the imperfection of subnetwork injection decisions, as discussed in Section 5.2.

We use deep router pipelines in this study because of their ubiquitous usage. Routers with shallow pipelines would activate VCs in a similar manner because the look-ahead signal to activate downstream drowsy SRAMs is generated with each switch allocation grant, which all routers with a switch have. To activate channels and lanes, ABNs with shallow router pipelines can either use predictors similar to non-drowsy SRAMs [33], or leave a small number of lanes constantly activated in order to better tolerate the activation delay of other lanes. In addition, ABNs have similar power gating granularity and therefore do not require more complicated power distribution networks or power gating transistors than multinets or other past work [26, 38, 12].

Future work for ABNs includes developing more sophisticated channel and switch lane activation policies. Policies that are aware of temporal and spatial locality of application traffic or maintain history of recent traffic characteristics can better predict the required bandwidth every cycle.

7. Conclusion

This paper proposes ABNs. ABNs divide channels and switches into lanes each of which can be activated and deactivated individually to match traffic demands. Unlike power-gating approaches with multiple subnetworks, flits are free to choose a different lane at each hop instead of committing to a set of lanes at injection time. With ABNs, on-chip network designers can design networks for worst-case traffic demands, without the disadvantage of unnecessarily incurring high static power overheads during periods of low or average activity. In the input buffer side, we take advantage of drowsy SRAM cells to activate and deactivate VCs individually without the possibility of false activations. ABNs also readily apply to silicon defect tolerance with just the extra cost for detecting faults. As we show for application traffic, ABNs reduce total power consumption by an average of 45% with comparable performance compared to single-lane power-gated networks. Compared to multi-network designs, ABNs reduce total power consumption by 33% with comparable or superior performance.

Acknowledgments

This work was supported by the Director, Office of Science, of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.

Disclaimer

Copyright Notice

References

[1] J. Balfour and W. J. Dally, "Design tradeoffs for tiled CMP on-chip networks," in *ICS '06: Proceedings of the 20th annual International Conference on Supercomputing*, 2006.

[2] N. Barrow-Williams, C. Fensch, and S. Moore, "A communication characterisation of Splash-2 and Parsec," in *Proceedings of the 2009 IEEE International Symposium on Workload Characterization (IISWC)*, ser. IISWC '09, 2009, pp. 86–97.

[3] D. U. Becker and W. J. Dally, "Allocator implementations for network-on-chip routers," in *Proceedings of the 2009 ACM/IEEE Conference on Supercomputing*, 2009.

[4] K. Bergman, S. Borkar, D. Campbell, W. Carlson, W. Dally, M. Denneau, P. Franzon, W. Harrod, J. Hiller, S. Karp, S. Keckler, D. Klein, R. Lucas, M. Richards, A. Scarpelli, S. Scott, A. Snavely, T. Sterling, R. S. Williams, K. Yelick, K. Bergman, S. Borkar, D. Campbell, W. Carlson, W. Dally, M. Denneau, P. Franzon, W. Harrod, J. Hiller, S. Keckler, D. Klein, P. Kogge, R. S. Williams, and K. Yelick, "Exascale computing study: Technology challenges in achieving exascale systems," http://users.ece.gatech.edu/~mrichard/ExascaleComputingStudyReports/exascale_final_report_100208.pdf, 2008.

[5] C. Bienia, "Benchmarking modern multiprocessors," Ph.D. dissertation, Princeton University, January 2011.

[6] J. Camacho and J. Flich, "HPC-mesh: A homogeneous parallel concentrated mesh for fault-tolerance and energy savings," in *Architectures for Networking and Communications Systems (ANCS), 2011 Seventh ACM/IEEE Symposium on*, 2011, pp. 69–80.

[7] C. Chen, Y. Lu, and S. D. Cotofana, "A novel flit serialization strategy to utilize partially faulty links in networks-on-chip," in *Proceedings of the 2012 IEEE/ACM Sixth International Symposium on Networks-on-Chip*, ser. NOCS '12, 2012, pp. 124–131.

[8] L. Chen and T. M. Pinkston, "Nord: Node-router decoupling for effective power-gating of on-chip routers," in *Proceedings of the 2012 45th Annual IEEE/ACM International Symposium on Microarchitecture*, ser. MICRO '12, 2012, pp. 270–281.

[9] X. Chen and L.-S. Peh, "Leakage power modeling and optimization in interconnection networks," in *Proceedings of the 2003 international symposium on Low power electronics and design*, ser. ISLPED '03, 2003, pp. 90–95.

[10] W. J. Dally and B. Towles, "Route packets, not wires: On-chip interconnection networks," in *Proceedings of the 38th annual Design Automation Conference*, 2001.

[11] ——, *Principles and Practices of Interconnection Networks*. Morgan Kaufmann Publishers Inc., 2003.

[12] R. Das, S. Narayanasamy, S. K. Satpathy, and R. G. Dreslinski, "Catnap: Energy proportional multiple network-on-chip," in *Proceedings of the 40th annual international symposium on Computer architecture*, ser. ISCA '13, 2013.

[13] H. Esmaeilzadeh, E. Blem, R. St. Amant, K. Sankaralingam, and D. Burger, "Dark silicon and the end of multicore scaling," in *Proceedings of the 38th annual international symposium on Computer architecture*, ser. ISCA '11, 2011, pp. 365–376.

[14] K. Flautner, N. S. Kim, S. Martin, D. Blaauw, and T. Mudge, "Drowsy caches: simple techniques for reducing leakage power," in *Proceedings of the 29th annual international symposium on Computer architecture*, ser. ISCA '02, 2002, pp. 148–157.

[15] A.-A. Ghofrani, R. Parikh, S. Shamshiri, A. DeOrio, K.-T. Cheng, and V. Bertacco, "Comprehensive online defect diagnosis in on-chip networks," in *VLSI Test Symposium (VTS), 2012 IEEE 30th*, 2012, pp. 44–49.

[16] P. V. Gratz and S. W. Keckler, "Realistic workload characterization and analysis for networks-on-chip design," in *4th Workshop on Chip Multiprocessor Memory Systems and Interconnects*, 2010.

[17] K. C. Hale, B. Grot, and S. W. Keckler, "Segment gating for static energy reduction in networks-on-chip," in *Proceedings of the 2nd International Workshop on Network on Chip Architectures*, ser. NoCArc '09, 2009, pp. 57–62.

[18] R. Hesse, J. Nicholls, and N. E. Jerger, "Fine-grained bandwidth adaptivity in networks-on-chip using bidirectional channels," in *Proceedings of the 2012 IEEE/ACM Sixth International Symposium on Networks-on-Chip*, ser. NOCS '12, 2012, pp. 132–141.

[19] J. Hestness, B. Grot, and S. W. Keckler, "Netrace: dependency-driven trace-based network-on-chip simulation," in *Proceedings of the Third International Workshop on Network on Chip Architectures*, ser. NoCArc '10. New York, NY, USA: ACM, 2010, pp. 31–36.

[20] R. R. Hoare, Z. Ding, and A. K. Jones, "A near-optimal real-time hardware scheduler for large cardinality crossbar switches," in *Proceedings of the 2006 ACM/IEEE Conference on Supercomputing*, 2006.

[21] Y. Hoskote, S. Vangal, A. Singh, N. Borkar, and S. Borkar, "A 5-GHz mesh interconnect for a teraflops processor," *Micro, IEEE*, vol. 27, no. 5, pp. 51–61, 2007.

[22] N. Jiang, D. Becker, G. Michelogiannakis, J. Balfour, B. Towles, D. Shaw, J. Kim, and W. Dally, "A detailed and flexible cycle-accurate network-on-chip simulator," in *IEEE International Symposium on Performance Analysis of Systems and Software*, ser. SPASS '13, 2013, pp. 86–96.

[23] A. Kahng, B. Li, L.-S. Peh, and K. Samadi, "ORION 2.0: A power-area simulator for interconnection networks," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 20, no. 1, pp. 191–196, 2012.

[24] H. Kaul, M. Anders, S. Hsu, A. Agarwal, R. Krishnamurthy, and S. Borkar, "Near-threshold voltage (NTV) design; opportunities and challenges," in *49th ACM/EDAC/IEEE Design Automation Conference (DAC)*, ser. DAC '12, 2012, pp. 1149–1154.

[25] H. S. Kia and C. Ababei, "Improving fault tolerance of network-on-chip links via minimal redundancy and reconfiguration," in *Proceedings of the 2011 International Conference on Reconfigurable Computing and FPGAs*, ser. RECONFIG '11, 2011, pp. 363–368.

[26] G. Kim, J. Kim, and S. Yoo, "Flexibuffer: reducing leakage power in on-chip network routers," in *Proceedings of the 48th Design Automation Conference*, ser. DAC '11, 2011, pp. 936–941.

[27] J. S. Kim, M. B. Taylor, J. Miller, and D. Wentzlaff, "Energy characterization of a tiled architecture processor with on-chip networks," in *Proceedings of the 2003 international symposium on Low power electronics and design*, ser. ISLPED '03, 2003, pp. 424–427.

[28] J. Kim, W. J. Dally, and D. Abts, "Flattened butterfly: a cost-efficient topology for high-radix networks," in *Proceedings of the 34th annual International Symposium on Computer Architecture*, 2007.

[29] M. Koibuchi, H. Matsutani, H. Amano, and T. M. Pinkston, "A lightweight fault-tolerant mechanism for network-on-chip," in *Proceedings of the Second ACM/IEEE International Symposium on Networks-on-Chip*, ser. NOCS '08, 2008, pp. 13–22.

[30] S. E. Lee and N. Bagherzadeh, "A variable frequency link for a power-aware network-on-chip (NoC)," *Integr. VLSI J.*, vol. 42, no. 4, pp. 479–485, 2009.

[31] H. Matsutani, M. Koibuchi, H. Amano, and D. Wang, "Run-time power gating of on-chip routers using look-ahead routing," in *Design Automation Conference, 2008. ASPDAC 2008. Asia and South Pacific*, 2008, pp. 55–60.

[32] H. Matsutani, M. Koibuchi, D. Ikebuchi, K. Usami, H. Nakamura, and H. Amano, "Ultra fine-grained run-time power gating of on-chip routers for CMPs," in *Networks-on-Chip (NOCS), 2010 Fourth ACM/IEEE International Symposium on*, 2010, pp. 61–68.

[33] H. Matsutani, M. Koibuchi, D. Wang, and H. Amano, "Adding slow-silent virtual channels for low-power on-chip networks," in *Networks-on-Chip, 2008. NoCS 2008. Second ACM/IEEE International Symposium on*, 2008, pp. 23–32.

[34] C. Meenderinck and B. Juurlink, "Euro-par 2008 workshops - parallel processing," E. César, M. Alexander, A. Streit, J. L. Träff, C. Cérin, A. Knüpfer, D. Kranzlmüller, and S. Jha, Eds., 2009, ch. (When) Will CMPs Hit the Power Wall?, pp. 184–193.

[35] G. Michelogiannakis, J. Balfour, and W. J. Dally, "Elastic buffer flow control for on-chip networks," in *HPCA '09: Proceeding of the 15th International Symposium on High-Performance Computer Architecture*, 2009, pp. 151–162.

[36] G. Michelogiannakis, D. Sanchez, W. J. Dally, and C. Kozyrakis, "Evaluating bufferless flow control for on-chip networks," in *NOCS '10: Proceedings of the Fourth International Symposium on Networks-on-Chip*, 2010, pp. 9–16.

[37] A. K. Mishra, A. Yanamandra, R. Das, S. Eachempati, R. Iyer, N. Vijaykrishnan, and C. R. Das, "RAFT: A router architecture with frequency tuning for on-chip networks," *Journal on Parallel and Distributed Computing*, vol. 71, no. 5, pp. 625–640, 2011.

[38] C. Nicopoulos, A. Yanamandra, S. Srinivasan, N. Vijaykrishnan, and M. Irwin, "Variation-aware low-power buffer design," in *Signals, Systems and Computers, 2007. ACSSC 2007. Conference Record of the Forty-First Asilomar Conference on*, 2007, pp. 1402–1406.

[39] M. Palesi, S. Kumar, and V. Catania, "Leveraging partially faulty links usage for enhancing yield and performance in networks-on-chip," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 29, no. 3, pp. 426–440, 2010.

[40] S. Rodrigo, J. Flich, A. Roca, S. Medardoni, D. Bertozzi, J. Camacho, F. Silla, and J. Duato, "Addressing manufacturing challenges with cost-efficient fault tolerant routing," in *Proceedings of the 2010 Fourth ACM/IEEE International Symposium on Networks-on-Chip*, ser. NOCS '10, 2010, pp. 25–32.

[41] D. Sanchez, G. Michelogiannakis, and C. Kozyrakis, "An analysis of interconnection networks for large scale chip-multiprocessors," *ACM Transactions on Architecture and Code Optimization*, vol. 7, no. 1, pp. 4:1–4:28, 2010.

[42] N. Seki, L. Zhao, J. Kei, D. Ikebuchi, Y. Kojima, Y. Hasegawa, H. Amano, T. Kashima, S. Takeda, T. Shirai, M. Nakata, K. Usami, T. Sunata, J. Kanai, M. Namiki, M. Kondo, and H. Nakamura, "A fine-grain dynamic sleep control scheme in MIPS R3000," in *Computer Design, 2008. ICCD 2008. IEEE International Conference on*, 2008, pp. 612–617.

[43] J. Shalf, S. S. Dosanjh, and J. Morrison, "Exascale computing technology challenges," in *International Meeting on High Performance Computing for Computational Science*, ser. VECPAR '10, 2010.

[44] L. Shang, L.-S. Peh, and N. K. Jha, "Dynamic voltage scaling with links for power optimization of interconnection networks," in *Proceedings of the 9th International Symposium on High-Performance Computer Architecture*, ser. HPCA '03, 2003, pp. 91–.

[45] V. Soteriou and L.-S. Peh, "Dynamic power management for power optimization of interconnection networks using on/off links," in *High Performance Interconnects, 2003. Proceedings. 11th Symposium on*, 2003, pp. 15–20.

[46] ——, "Exploring the design space of self-regulating power-aware on/off interconnection networks," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 18, no. 3, pp. 393–408, 2007.

[47] W.-C. Tsai, D.-Y. Zheng, S.-J. Chen, and Y.-H. Hu, "A fault-tolerant noc scheme using bidirectional channel," in *Design Automation Conference (DAC), 2011 48th ACM/EDAC/IEEE*, 2011, pp. 918–923.

[48] K. Usami and N. Ohkubo, "A design approach for fine-grained run-time power gating using locally extracted sleep signals," in *Computer Design, 2006. ICCD 2006. International Conference on*, 2006, pp. 155–161.

[49] H. Wang, L.-S. Peh, and S. Malik, "Power-driven design of router microarchitectures in on-chip networks," in *Proc. of the 36th annual IEEE/ACM Intl. Symp. on Microarchitecture*, 2003.

[50] L. Wang, P. Kumar, K. H. Yum, and E. J. Kim, "APCR: an adaptive physical channel regulator for on-chip interconnects," in *Proceedings of the 21st international conference on Parallel architectures and compilation techniques*, ser. PACT '12, 2012, pp. 87–96.

[51] J. Xu, W. Wolf, J. Henkel, and S. Chakradhar, "A design methodology for application-specific networks-on-chip," *ACM Transactions on Embedded Computer Systems*, vol. 5, no. 2, pp. 263–280, 2006.

[52] Q. Yu and P. Ampadu, "Transient and permanent error co-management method for reliable networks-on-chip," in *Proceedings of the 2010 Fourth ACM/IEEE International Symposium on Networks-on-Chip*, ser. NOCS '10, 2010, pp. 145–154.